# *Data Pictorial*: Deconstructing Raster Images for Data-Aware Animated Vector Posters


Tongyu Zhou
Brown University
Providence, RI, USA

Gromit Yeuk-Yin Chan
Adobe Research
San Jose, CA, USA

Shunan Guo
Adobe Research
San Jose, CA, USA

Jane Hoffswell
Adobe Research
San Jose, CA, USA

Chang Xiao
Adobe Research
San Jose, CA, USA

Victor Soares Bursztyn
Adobe Research
San Jose, CA, USA

Eunyee Koh
Adobe Research
San Jose, CA, USA


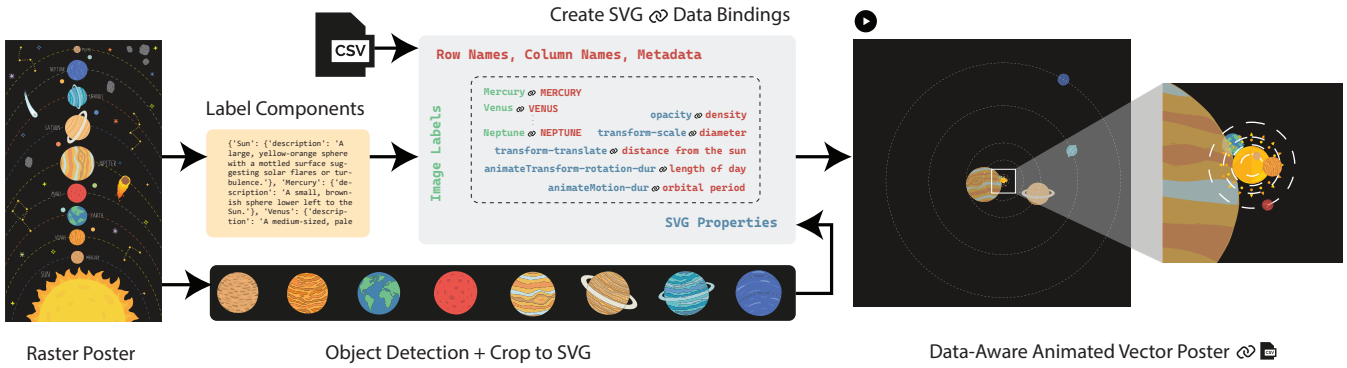


**Figure 1: Our pipeline for Data Pictorial: a static raster poster is decomposed into cropped, labeled components that are converted to SVG elements. Bindings are then established between the metadata of an input data table, the image labels, and SVG properties to construct a data-aware vectorized poster that dynamically cycles through data values via SMIL animation.**


## ABSTRACT

To support data integration into pictorials, we propose Data Pictorial, a pipeline that deconstructs a raster image into SVG objects whose attributes are contextualized in data. This process is achieved by cropping objects of interest using zero-shot detection, converting them into quantized bitmaps, and tracing the results as SVG paths. The technique then provides suggestions for binding the SVG objects and properties with data fields, affording the flexibility to automatically modify and animate the SVG based on the mapping. The resultant data-aware vector hypermedia can be potential candidates for real-time data inspection and personalization, all while maintaining the aesthetic of the original pictorial.



## CCS CONCEPTS

• **Human-centered computing** → **Visualization toolkits**.

## KEYWORDS

lazy data binding, infographics, animated vector graphics




*UIST Adjunct '24, October 13–16, 2024, Pittsburgh, PA, USA*



ACM ISBN 979-8-4007-0718-6/24/10
https://doi.org/10.1145/3672539.3686353

**ACM Reference Format:**
Tongyu Zhou, Gromit Yeuk-Yin Chan, Shunan Guo, Jane Hoffswell, Chang Xiao, Victor Soares Bursztyn, and Eunyee Koh. 2024. *Data Pictorial*: Deconstructing Raster Images for Data-Aware Animated Vector Posters. In *The 37th Annual ACM Symposium on User Interface Software and Technology (UIST Adjunct '24), October 13–16, 2024, Pittsburgh, PA, USA.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3672539.3686353


## 1 INTRODUCTION

Pictorial posters, as a subcategory of infographics, convey key insights to a viewer through visual illustrations. However, they often do not depict the actual data backing their messages, and can be tedious to modify if the user wishes to inject this data while maintaining the style of the pictorial. Recent approaches in generating pictorial visualizations such as DataQuilt [16], MetaGlyph [15], ChartSpark [14], and Infomages [2] have thus proposed pipelines where images are processed as backgrounds, glyphs, or stylistic guides for different types of visualizations. However, these workflows assume a template visualization such as a bar graph, line chart, or scatterplot as the primary subject to guide the layout of the visual elements. The pictorial illustrations either lose their original layouts in favor of the new layout [16], or must have an original layout that resembles the desired visualization layout [2]. Other similar work in lazy data-binding with expressive graphics [4, 6, 13] are not restrictive to visualization layouts, but either require the user to author vector graphics within the system or import a suitable one that contains elements that can be bound to data.

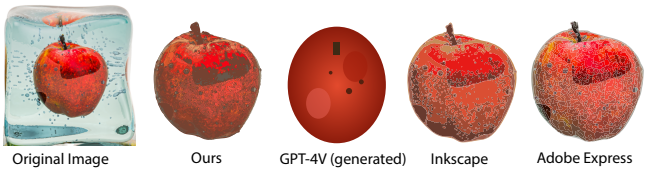


**Figure 2: Comparison of our raster to SVG approach, Data Pictorial, to existing conversion methods [1, 3, 9].**

In contrast, our proposed approach allows the user to bind data to any raster image by deconstructing it into vectorized SVG components. We compare our conversion strategy against existing commercial methods [1, 3, 9], noting that more complex models such as DiffVG [5], LIVE [7], or Neural Painting [17] may provide greater fine-tuning at the expense of simplicity. Our pipeline, Data Pictorial, then recommends bindings between properties of the vector objects and the input data fields. While prior work has focused only on static data bindings, Data Pictorial allows data fields to be directly bound to *animated* parameters in addition to static ones.

## 2 DATA PICTORIAL

### 2.1 Deconstructing Raster Posters

We segment the raster-based image using zero-shot object detection with the pretrained OWL-ViT model [8] to identify objects to convert into SVG elements. The background of the cropped image is removed using `rembg`, a port of $U^2$-Net [10]. Color quantization is then performed on the image to split it into $n$ colors as specified by the user (with a default value of $n = 10$). A binary mask is created for each color; we then apply a Gaussian blur, with user-adjustable standard deviation, to the mask for antialiasing. The blurred mask is used to extract bitmaps for each color from the original image. Each bitmap is passed to `potrace` [11], a polygon-based tracing algorithm, to extract Bézier curves to convert into SVG path commands. All SVGs paths are then combined to create the SVG object. Figure 2 shows an example output of this conversion process compared to existing raster-to-SVG pipelines. We note that generative vision models like GPT4 (2024-02-15-preview) are not yet able to handle complex SVGs, and other approaches may result in artifacts.

### 2.2 Mapping SVG Properties to Data

To associate mappings between data and the now converted SVG properties, we assume we have a CSV as our input dataset. We then prompt GPT4V to label all the objects in the original image with a one-sentence description of each object's visual appearance. We also gather a list of SVG properties such as $x$ position, $y$ position, rotation, scale, opacity, etc. With this information, we create embeddings for the row and column names of the dataset, the label and descriptions, and SVG property names using the sentence transformers model MiniLM [12]. We obtain the most similar pairs of embeddings by computing pairwise cosine similarities. The user can use these recommendations, or directly assign the desired mapping.

*2.2.1* ***Static Attributes.*** We create mappings between data fields and the static attributes of SVG elements by scaling the relevant data fields to the appropriate ranges supported by the SVG attribute.

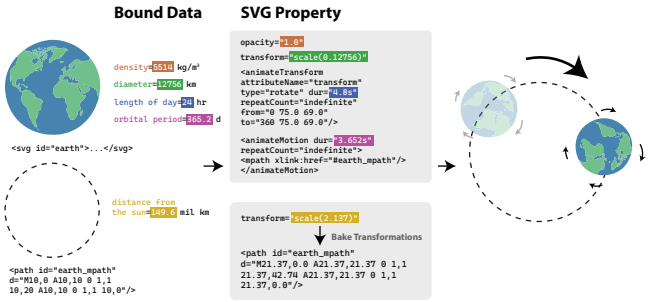


**Figure 3: Data fields and SVG properties are bound by generating attribute strings with the scaled data for each SVG.**

For example, to map density ($d \in [687, 5514]$ based on densities of all the planets) to opacity ($o \in [0, 1]$), the density must be re-scaled to that range, $o = (d - d_{min}) * (o_{max} - o_{min})/(d_{max} - d_{min})$. Users can additionally enter custom minimum $o_{min}$ and maximum $o_{max}$ ranges for each SVG attribute to create particular visual effects or if the attribute does not have set ranges (i.e., duration). Then, for each SVG object, we generate an attribute string corresponding to each data field (Figure 3) and merge the strings accounting for overlaps in attributes, resulting in a final attribute string that can be parsed by any modern web browser. Each SVG object is converted to a `<group>` element and appended to a larger SVG wrapper.

*2.2.2* ***Animations.*** We support two types of data-aware animations, 1) *in-situ* and 2) *motion-path*. In-situ animations transform the SVG object in place, and are constructed similarly to the static attributes. The only difference is that instead of modifying the attributes, we add `<animate>` and `<animateTransform>` child elements to the SVG element and map data properties to these animated attributes. Conversely, motion-path animations move SVG objects over a given trajectory, which can be extracted from any closed path (curve or line) from the original static raster image or additionally drawn by the user. SVG objects are then instructed to follow these paths by referencing them with `<animateMotion>` and `<mpath>` tags. Per the SVG specification, path following only works for the absolute coordinates of the path, specified by the `d` attribute, so we automatically bake transformation matrices into the path when it is moved, scaled, or rotated.

## 3 APPLICATIONS & FUTURE WORK

Future work will focus on an accompanying direct manipulation interface for users, as well as a study to evaluate the possible interactions and usability of the technique. Additionally, data in the resultant SVGs can be updated in real time by embedding JavaScript to pull from data APIs or system time when loaded. However, we note that further work is needed to handle instances where the structure of the dataset changes. While Data Pictorial focuses on binding pictorials to standard datasets, other potential use cases include extending this technique to other types of visualizations and personalization–empowering the user to attribute SVG properties to individual traits when sharing vector posters with others.

# REFERENCES


[1] Adobe Inc. 2015. Adobe Express. https://www.adobe.com/express/feature/image/convert/png-to-svg

[2] D. Coelho and K. Mueller. 2020. Infomages: Embedding Data into Thematic Images. *Computer Graphics Forum* 39, 3 (2020), 593–606. https://doi.org/10.1111/cgf.14004 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.14004

[3] Inkscape Project. 2003. Inkscape. https://inkscape.org/

[4] Nam Wook Kim, Eston Schweickart, Zhicheng Liu, Mira Dontcheva, Wilmot Li, Jovan Popovic, and Hanspeter Pfister. 2017. Data-Driven Guides: Supporting Expressive Design for Information Graphics. *IEEE Transactions on Visualization and Computer Graphics* 23, 1 (2017), 491–500. https://doi.org/10.1109/TVCG.2016.2598620

[5] Tzu-Mao Li, Michal Lukáč, Michaël Gharbi, and Jonathan Ragan-Kelley. 2020. Differentiable vector graphics rasterization for editing and learning. *ACM Transactions on Graphics (TOG)* 39, 6 (2020), 1–15.

[6] Zhicheng Liu, John Thompson, Alan Wilson, Mira Dontcheva, James Delorey, Sam Grigg, Bernard Kerr, and John Stasko. 2018. Data Illustrator: Augmenting Vector Design Tools with Lazy Data Binding for Expressive Visualization Authoring. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (, Montreal QC, Canada,) *(CHI '18)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3173574.3173697

[7] Xu Ma, Yuqian Zhou, Xingqian Xu, Bin Sun, Valerii Filev, Nikita Orlov, Yun Fu, and Humphrey Shi. 2022. Towards Layer-wise Image Vectorization. arXiv:2206.04655 [cs.CV]

[8] Matthias Minderer, Alexey Gritsenko, Austin Stone, Maxim Neumann, Dirk Weissenborn, Alexey Dosovitskiy, Aravindh Mahendran, Anurag Arnab, Mostafa Dehghani, Zhuoran Shen, Xiao Wang, Xiaohua Zhai, Thomas Kipf, and Neil Houlsby. 2022. Simple Open-Vocabulary Object Detection with Vision Transformers. arXiv:2205.06230 [cs.CV]

[9] OpenAI. 2023. GPT4V. https://openai.com/index/gpt-4v-system-card/

[10] Xuebin Qin, Zichen Zhang, Chenyang Huang, Masood Dehghan, Osmar R. Zaiane, and Martin Jagersand. 2020. U2-Net: Going deeper with nested U-structure for salient object detection. *Pattern Recognition* 106 (Oct. 2020), 107404. https://doi.org/10.1016/j.patcog.2020.107404

[11] Peter Selinger. 2003. Potrace: a polygon-based tracing algorithm.

[12] Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. MiniLM: Deep Self-Attention Distillation for Task-Agnostic Compression of Pre-Trained Transformers. arXiv:2002.10957 [cs.CL]

[13] Haijun Xia, Nathalie Henry Riche, Fanny Chevalier, Bruno De Araujo, and Daniel Wigdor. 2018. DataInk: Direct and Creative Data-Oriented Drawing. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (, Montreal QC, Canada,) *(CHI '18)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3173574.3173797

[14] Shishi Xiao, Suizi Huang, Yue Lin, Yilin Ye, and Wei Zeng. 2024. Let the Chart Spark: Embedding Semantic Context into Chart with Text-to-Image Generative Model. *IEEE Transactions on Visualization and Computer Graphics* 30, 1 (2024), 284–294. https://doi.org/10.1109/TVCG.2023.3326913

[15] Lu Ying, Xinhuan Shu, Dazhen Deng, Yuchen Yang, Tan Tang, Lingyun Yu, and Yingcai Wu. 2023. MetaGlyph: Automatic Generation of Metaphoric Glyph-based Visualization. *IEEE Transactions on Visualization and Computer Graphics* 29, 1 (2023), 331–341. https://doi.org/10.1109/TVCG.2022.3209447

[16] Jiayi Eris Zhang, Nicole Sultanum, Anastasia Bezerianos, and Fanny Chevalier. 2020. DataQuilt: Extracting Visual Elements from Images to Craft Pictorial Visualizations. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (, Honolulu, HI, USA,) *(CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3313831.3376172

[17] Zhengxia Zou, Tianyang Shi, Shuang Qiu, Yi Yuan, and Zhenwei Shi. 2020. Stylized Neural Painting. arXiv:2011.08114 https://arxiv.org/abs/2011.08114